Scheduling for Distributed Sensor Networks

Vijay Gupta, Timothy Chung, Babak Hassibi, and Richard M. Murray

Division of Engineering and Applied Science, Caltech, Pasadena, CA 91125, USA,
{gupta,timothyc,hassibi,murray}@caltech.edu

Abstract. We examine the problem of distributed estimation when only one sensor can take a measurement per time step. The measurements are then exchanged among the sensors. The problem is motivated by the use of sonar range-finders used by the vehicles on the Caltech Multi-Vehicle Wireless Testbed. We solve for the optimal recursive estimation algorithm when the sensor switching schedule is given. Then we investigate several approaches for determining an optimal sensor switching strategy. We see that this problem involves searching a tree in general and propose and analyze two strategies for pruning the tree to keep the computation limited. The first is a sliding window strategy motivated by the Viterbi algorithm, and the second one uses thresholding. We also study a technique that employs choosing the sensors randomly from a probability distribution which can then be optimized. The performance of the algorithms are illustrated with the help of numerical examples.

1 Introduction and Motivation

Recently there has been a lot of interest in networks of sensing agents which act cooperatively to obtain the best estimate possible, (e.g. [1] and the references therein). While such a scheme admittedly has higher complexity than the strategy of treating each sensor independently, the increased accuracy often makes it worthwhile. If all the sensors exchange their measurements, the resulting estimate can be better even than the sensor with the least measurement noise (were no information exchange happening). The advantages of forming sensor networks are even greater if the sensors are heterogenous. The increased complexity arises from the communication infrastructure atop every sensor and the algorithmic changes needed for fusing the measurements from other sensors to obtain a better estimate.

Because of the above-mentioned advantages, there has been a lot of attention on data fusion of heterogeneous sensor measurements, as in [2]. Works such as the EYES project [3], WINS [4], and Smart Dust [5], are examples of systems implementing such networks. The assumption usually made in the analysis of such systems is that all the sensors take measurements at the same time. Thus the main issue is multi-sensor data fusion. One example of many sensor fusion algorithms can be found in [6]. The sensor management issues, if present at all are in the context of energy efficiency [7, 8], imperfect localization of sensor platforms [9], optimal coverage of a given region [10, 9], and efficient networking and communication protocols [11].

However, in some applications, the use of one sensor places restrictions on the use of other sensors. This situation exists whenever simultaneous use of sensors causes interference in measurements. We face this situation in our own work related to the Caltech Multi-Vehicle Wireless Testbed (MVWT) [12]. When the individual vehicles are using sonar range-finding devices, only one sensor can be active at any time. In such a case, apart from the issue of optimal multi-sensor data fusion, there is the additional issue of optimally scheduling the sensor measurements so as to minimize the state estimate error covariance.

In this paper, we study this problem of coming up with the optimal sensor schedule when only one sensor is allowed to take the measurement at every time step. While optimization of sensor schedules have been examined using optimal or stochastic control theory techniques, as in [13, 14], solutions to Ricatti differential equations, and even information-theoretic methods, as in [15], we pursue two simpler methods, sliding window and thresholding, for determining an optimal sensing schedule. These methods trade computation/memory requirements for sub-optimality; however, they seem to work well on the simulation examples. In addition, we also study a method that involves simply choosing the sensors randomly according to some probability distribution. The probability distribution can then be optimally chosen so as to minimize the expected error covariance.

The paper is organized as follows. In the next section, we set up the problem and solve for the optimal data-fusion algorithm for a given sensor schedule. We briefly consider the degradation in the performance when this scheme is used for the case when communication noise is present. Then we consider the question of choosing the optimal sensor schedule, which is the focus of the paper. We present some methods that obtain sub-optimal sensor schedules, but have the advantage of being much simpler to use. We demonstrate these algorithms with the help of examples and end with conclusions and scope for future work.

2 Modeling and Problem Formulation

Consider the system evolving as follows.

$$x[k+1] = Ax[k] + Bw[k]. \tag{1}$$

$x[k] \in \mathbf{R}^n$ is the process state at time step k and $w[k]$ is the process noise. The process noise is assumed white, Gaussian and zero mean with covariance matrix Q. The process state is being observed by N sensors with the measurement equation for the i-th sensor being

$$y_i[k] = C_i x[k] + v_i[k], \tag{2}$$

where $y_i[k] \in \mathbf{R}^s$ is the measurement. The measurement noises $v_i[k]$'s for the sensors are assumed independent of each other and of the process noise. Further the noise $v_i[k]$ is assumed to be white, Gaussian and zero mean with covariance matrix R_i. It is assumed that only one sensor can be used at any time. However, unless stated otherwise, we assume that the measurements are communicated

to all the sensors in an error-free manner. The estimate of i-th sensor given the measurements till time steps $k-1$ is denoted by $\hat{x}_i[k|k-1]$ or in short as $\hat{x}_i[k]$. We first pose the question: Assuming the sensor switching sequence to be given what is the optimal filtering for the i-th node?

It is fairly obvious that the innovation for the i-th node at time step k is given by

$$e_i[k] = y_j[k] - C_j\hat{x}_i[k|k-1], \tag{3}$$

where we have assumed that the j-th sensor takes the measurement at time step k. Then following the standard derivation (see, e.g., [16]), we obtain the recursive optimal filtering equation as

$$\hat{x}_i[k+1|k] = A\hat{x}_i[k|k-1] + K_k^i e_i[k],$$

where

$$K_k^i = AP_i[k|k-1]C_j^T R_{e_j[k]}^{-1}, \qquad R_{e_i[k]} = C_j P_i[k|k-1]C_j^T + R_j$$

and $P_i[k|k-1]$'s evolve as

$$P_i[k+1|k] = (A - K_k^i C_j)P_i[k|k-1](A - K_k^i C_j)^T + BQB^T + K_k^i R_j (K_k^i)^T. \tag{4}$$

Assuming the initial state $x[0]$ has mean zero and covariance Π_0, the initial covariance matrix for above recursions is also given by $P_i(0|-1) = \Pi_0$. $P_i[k|k-1]$ is the error covariance for the i-th sensor at time step k when it has processed the measurements till time step $k-1$. We will refer to it as $P_i[k]$ in short. Note that since all the nodes have access to the same measurements, then there is only one innovation and hence all the state estimates are the same. So the subscript i is unnecessary in this case and $P_i[k] = P[k]$ for all i. We find that the optimal equations are simply those of a Kalman filter assuming a time-varying sensor. This is not surprising since all the measurements are being shared and hence the underlying philosophy is still that of a centralized estimator.

2.1 Kalman Filter - Communication Noise

Let us assume now that any signal exchanged between sensor nodes i and j is corrupted by additive, zero-mean, Gaussian white noise, v_{ij}. We wish to see how the performance of the scheme of exchanging measurements between the sensors outlined above is affected. Going through a similar derivation as above, we find that equation (3) is modified to

$$e_i[k] = y_j[k] - C_j\hat{x}_i[k|k-1] + v_{ij}[k],$$

assuming that j-th sensor has taken the measurement at time step k. Let us assume the noise vector $\zeta[k] = (w[k], v_i[k], v_{ij}[k])^T$ to be described by

$$E\left[\zeta[k]\zeta[l]^T\right] \triangleq \begin{pmatrix} Q & 0 & 0 \\ 0 & R_i & 0 \\ 0 & 0 & R_{ij} \end{pmatrix} \delta(k-l).$$

Then, we find that the Kalman filter form remains the same as before except that equation (2) becomes

$$R_{e_i[k]} = C_j P_i[k|k-1] C_j^T + R_j + R_{ij}. \tag{5}$$

and equation (4) changes to

$$\begin{aligned} P_i[k+1|k] = (A - K_k^i C_j) P_i[k|k-1] (A - K_k^i C_j)^T + BQB^T + K_k^i R_j (K_k^i)^T \\ + K_k^i R_{ij} (K_k^i)^T \end{aligned} \tag{6}$$

We note that the only difference from the earlier case is that the effective measurement noise includes the actual sensor noise plus the communication noise. Observe, however, that sending only the measurement from one sensor to the other might not be the optimal thing to do in this case. Sending more information (e.g., the state estimates) might lead to better performance for all the sensors considered together.

2.2 Optimization of the Sensor Schedule

In the analysis presented so far, we have assumed that the sensor schedule was given. Thus we wanted to find out the optimal recursive solution to the problem of optimal estimation for a fixed sensor switching sequence. It is obvious that the minimum error covariance achievable is a function of the sensor schedule. Thus, a more general problem is that of finding the optimal switching sequence. We wish to find the sensor schedule that minimizes the error covariance for the sensors over a given time horizon. In the next section, we consider this problem. For simplicity and without loss of generality, we consider only two sensors and define the cost function to be the sum of the error covariance matrices for the two sensors over the running time of the system. In other words, our cost function J is given by

$$J = \sum_{k=0}^{N} \text{trace}\left(P_1[k] + P_2[k]\right),$$

where, as before, $P_1[k]$ and $P_2[k]$ are error covariances of the estimates of the two sensors at time step k. We have assumed that the system begins at time $k = 0$ and goes on till $k = N$. In a more general case, the covariances can be variously weighted to set up the cost function if getting a good estimate either at some time steps or for some sensors is more important than others.

3 Optimization Algorithms

We can represent all the possible sensor schedule choices by a tree structure, as shown in figure 1 for the case of two sensors. The depth of any node in the tree represents time instants with the root at time zero. The branches correspond to choosing a particular sensor to be active at that time instant. Thus, the path from the root to any node at depth d represents a particular sensor schedule

choice for time steps 0 to d. We can associate with each node the cost function evaluated using the sensor schedule corresponding to the path from the root to that node. Obviously, finding *the* optimal sequence requires traversing all the paths from the root to the leaves in a binary tree (for the case of two sensors). If the leaves are at a depth N, a total of 2^N schedules need to be compared. This procedure might place too high a demand on the computational and memory resources of the system. Moreover, in practical applications N might not be fixed a-priori. Hence we need some sort of on-line optimization procedure. We present some approximations which address these difficulties.

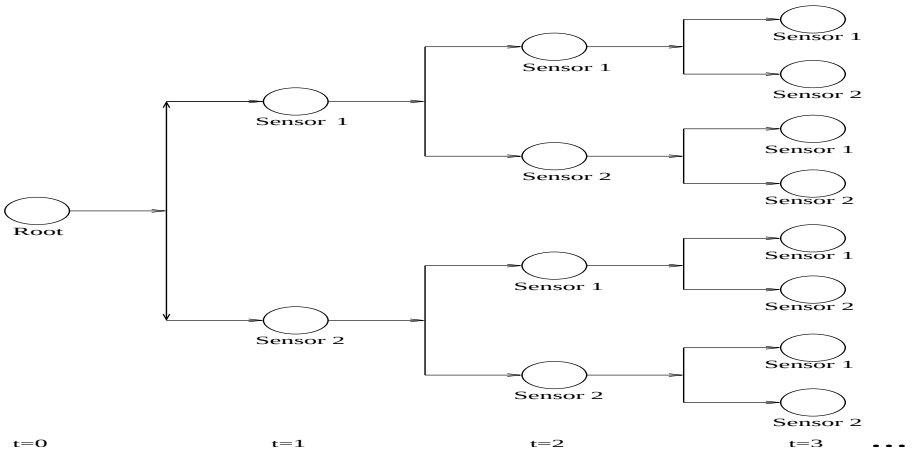

Fig. 1. The tree structure defined by the various possible choices of sensor schedules illustrated for the case of 2 sensors.

The first two approximations aim at pruning the tree so as to keep it to a manageable size. They both involve choosing some arbitrary parameters which depend on the problem and the computation/memory resources available. Choosing these parameters conservatively will ensure that our sub-optimal solution will be closer to the optimal solution but it might mean keeping a large part of the tree intact. Thus there is a trade-off involved. However, in the numerical examples studied, relatively liberal choices still kept the tree size fairly small. The third algorithm we present aims at doing away with traversing the tree altogether although it tries to minimize only the expected steady state error covariance matrices. We now consider these three schemes.

3.1 Sliding Window Algorithm

This algorithm is similar to a pseudo real time version of the Viterbi algorithm. We define a window size d where $d < N$. The algorithm proceeds as follows:

1. Initialization: Start from root node with time $k = 0$.

2. Traversal:
   (a) Traverse all the possible paths in the tree for the next d levels from the present node.
   (b) Identify the sensor sequence S_k,S_{k+1},S_{k+2},..., S_{k+d-1} that yields the minimum cost at the end of this window of size d.
   (c) Choose the first sensor S_k from the sequence.
3. Sliding the Window:
   (a) If $k = N$ then quit, else go to the next step.
   (b) Designate the sensor S_k as the root.
   (c) Update time $k = k + 1$.
   (d) Repeat the traversal step.

The arbitrary parameter for this algorithm, mentioned earlier, is the window size d. If the window size is large enough, the sequence yielding the lowest cost will resemble the optimal sequence for the entire time horizon. Also note that when we slide the window, we already have the error covariances for the first $d - 1$ time steps stored; hence they do not need to be recalculated. Consequently, the method is not very computationally intensive.

3.2 Thresholding

This algorithm is similar to that presented in [17], in the context of choosing the optimal controller from a set of many possible choices. We define a factor f where $f \geq 1$. The algorithm proceeds as follows:

1. Initialization : Start from root node with cost 0.
2. Pruning:
   (a) Traverse the tree by one step through all possible paths from the current node.
   (b) Calculate the minimum cost till that time step.
   (c) Prune away any branch that yields the cost greater than f times the minimum.
   (d) For the remaining branches, denote the cost of the nodes as the cost achieved by moving down the tree till the node.
3. Update: Consider each node in the next time step as the root node and repeat the pruning step.
4. After N time steps or a sufficiently large time interval, declare the optimal sequence to be the one yielding the minimum cost till that time step.

The intuition behind the method is that any sequence which yields too high a cost at any intermediate time step would probably not be the one that yields the minimum cost over-all. By playing with the factor f, we obtain a trade-off between the certainty that we would not prune away the optimal sequence and the number of branches of the error covariance tree that need to be traversed.

3.3 Randomly Chosen Sensors

This algorithm aims to do away with traversing the tree altogether. The cost function for this algorithm is the steady state error covariance for the two sensors. In this algorithm, the sensors are chosen randomly according to some probability distribution. The choice is done independently at every time step. The probability distribution is then chosen so as to minimize the expected steady state error covariance. Note that we can not calculate the exact value of the error covariance since that will depend on the specific sensor schedule chosen. Hence we optimize the expected value of the error covariance. For obtaining the expected value given any particular probability distribution, we proceed as follows.

Consider the time-varying Kalman filter recursion for the system given by the equation (1). The measurement equation is given by equation (2). The sensor (or in other words, the observation matrix C) at every time step k is chosen independently from among the choices $C_1, C_2, \ldots, C_N$. The associated sensor noise covariances are $R_1, R_2, \ldots, R_N$. The choice at every time step is independent from choices at other time steps. Further the probability of C_i being chosen at any time step is q_i, which remains constant with time. Thus the Riccati recursion for the error covariance for any sensor is given by

$$\begin{aligned} P[k+1] = BQB^T + AP[k]A^T \\ -AP[k]C[k]^T \left(R[k] + C[k]P[k]C[k]^T\right)^{-1} C[k]P[k]A^T, \end{aligned}$$

with $P[0]$ as the initial condition. The quantity $R[k]$ in the above equation is the sensor noise that depends on the particular sensor chosen at time step k. This yields the quantity $P[k+1]$ as random since it depends on the particular sequence of chosen $C[i]$'s ($0 \leq i \leq k$). We look at its expected value and try to evaluate the limit of the expectation as $k \to \infty$.

Thus we are interested in

$$\begin{aligned} E\left[P[k+1]\right] = E\left[BQB^T + AP[k]A^T\right] \\ -E\left[AP[k]C[k]^T(R[k] + C[k]P[k]C[k]^T)^{-1}C[k]P[k]A^T\right] \quad (7) \end{aligned}$$

Explicitly evaluating this expectation appears to be intractable. We look instead for an upper bound. We proceed as follows. First note that the quantities $P[k]$ and $C[k]$ are independent. Thus we can explicitly take the expectation with respect to the probability distribution of $C[k]$ and write

$$\begin{aligned} E\left[P[k+1]\right] = BQB^T + AE\left[P[k]\right]A^T \\ -\sum q_i AE\left[P[k]C_i^T(R_i + C_iP[k]C_i^T)^{-1}C_iP[k]\right]A^T, \end{aligned}$$

where the expectations on the right hand side are now over $C[0], \ldots, C[k-1]$. Now note the following result.

Lemma 1. *$APC^T(R + CPC^T)^{-1}CPA^T$ is convex in P provided P is positive semi-definite and R is positive definite.*

Proof. We use the following fact [18]. A function $f(x)$ is convex in x if and only if $f(x_0 + th)$ is convex in the scalar t for all x_0 and h. Thus consider

$$\Sigma = A(P_0 + tZ)C^T \left(R + C(P_0 + tZ)C^T\right)^{-1} C(P_0 + tZ)A^T.$$

Calling $R + C(P_0 + tZ)C^T$ as X, we obtain

$$\begin{aligned}\frac{\partial \Sigma}{\partial t} &= AZC^T X^{-1} C(P_0 + tZ)A^T \\ &\quad + A(P_0 + tZ)C^T X^{-1} CZ \left[I - C^T X^{-1} C(P_0 + tZ)\right] A^T.\end{aligned}$$

Thus the second derivative is given by

$$\frac{\partial^2 \Sigma}{\partial t^2} = 2\Lambda X^{-1} \Lambda^T,$$

where

$$\Lambda = A\left[(P_0 + tZ)C^T X^{-1} CZC^T - ZC^T\right].$$

Note that X is positive definite under the stated conditions on P and R, hence X^{-1} exists and is positive definite. The second derivative is positive everywhere which proves the assertion. □

To evaluate the upper bound, we use Jensen's inequality (see, e.g., [19]).

Proposition 1 (Jensen's Inequality). *If a function $f(x)$ is convex, $f\left(E[x]\right) \leq E\left[f(x)\right]$.*

Using the above results, we immediately obtain

$$\begin{aligned}E\left[P[k+1]\right] &= BQB^T + AE\left[P[k]\right]A^T \\ &\quad - \sum q_i AE\left[P[k]C_i^T (R_i + C_i P[k] C_i^T)^{-1} C_i P[k]\right] A^T \\ &\leq BQB^T + AE\left[P[k]\right]A^T \\ &\quad - \sum q_i A\left[E[P[k]]C_i^T (R_i + C_i E[P[k]] C_i^T)^{-1} C_i E[P[k]]\right] A^T.\end{aligned} \tag{8}$$

This is an upper bound on the Riccati recursion in equation (7).

We now try to optimize the upper bound by choosing q_i's appropriately. First we need to check for the convergence of the above recursion for the upper bound as time progresses. We note the following convergence result that can be proved on the lines of [20].

Theorem 1. *Define*

$$g_Q(X) = AXA^T + BQB^T - \sum q_i AXC_i^T \left(C_i X C_i^T + R_i\right)^{-1} C_i X A^T,$$

and the operator

$$\phi(K_i, X) = \sum q_i \left(A_i X A_i^T + V_i\right),$$

where

$$A_i = A + K_i C_i, \qquad V_i = BQB^T + K_i R_i K_i^T.$$

Suppose there exist K_i, P *such that* $P > 0$ *and* $P > \phi(K_i, P)$*. Then the iteration*

$$P[k+1] = BQB^T + AP[k]A^T - \sum q_i A \left[P[k]C_i^T (R_i + C_i P[k] C_i^T)^{-1} C_i P[k]\right] A^T,$$

converges for all initial conditions $P[0] \geq 0$*. Further the limit* $\bar{P}$ *is the unique positive semi-definite solution of the equation*

$$X = BQB^T + AXA^T - \sum q_i A \left[XC_i^T (R_i + C_i X C_i^T)^{-1} C_i X\right] C_i^T. \qquad (9)$$

If A is stable, we can always find K_i, P satisfying the above conditions by choosing K_i as the zero matrices and P as $2\bar{P}$ where $\bar{P}$ is the positive definite solution of the Lyapunov equation

$$\bar{P} = A\bar{P}A^T + BQB^T.$$

Applying the above result to find the convergence of the recursion in equation (8), we see that as long as A is stable, the recursion converges and the expected value of error covariance is the unique positive semi-definite solution of the equation

$$X = BQB^T + AXA^T - \sum q_i A \left[XC_i^T (R_i + C_i X C_i^T)^{-1} C_i X\right] C_i^T.$$

Since A being stable is the chief concern in practical applications of estimation, we see that the recursion for the upper bound always converges and the solution can be obtained by solving the corresponding equation (9).

The algorithm thus consists of choosing q_i's so as to optimize the upper bound as a means of optimizing the expected steady state value of P_k itself. The optimization problem is solved under the constraint

$$\sum_{i=1}^{N} q_i = 1, \qquad q_i \geq 0, \qquad \forall i = 1, \ldots, N.$$

The problem can be solved by the gradient search algorithm or even by brute force search for small N.

4 Simulation Results

4.1 Example model and cost function

As pointed out earlier, the motivating example for this work was the use of sonar range finding sensors on vehicles on the MVWT testbed. In this section, we walk through an example demonstrating the use of algorithms developed above on such a system. We assume two sensing vehicles trying to locate a non-cooperating target. We model the target vehicle with the standard constant acceleration model [21]. This model assumes that the vehicle has constant acceleration equal

to zero except for a small perturbation. We assume that the vehicle moves in two dimensions. Denoting the position of the vehicle in the two dimensions by p_x and p_y, and the velocities by v_x and v_y, we can model the state of the system by the vector

$$X = \begin{bmatrix} p_x \\ p_y \\ v_x \\ v_y \end{bmatrix}.$$

With a discretization step size of h, the dynamics of the vehicle can be modeled as

$$X(k+1) = AX(k) + Bw(k), \tag{10}$$

where

$$A = \begin{bmatrix} 1 & 0 & h & 0 \\ 0 & 1 & 0 & h \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} h^2/2 & 0 \\ 0 & h^2/2 \\ h & 0 \\ 0 & h \end{bmatrix}.$$

The term w_k represents the noise that enters because of the perturbation in the zero accelerations in the two dimensions. The sensor model is the usual sonar model [22]. Being an echo-based device, it senses only the positions and not the velocities. If the sensor is oriented at an angle θ to the global x-axis (see figure 2), it can be shown ([22]) that the vehicle's measurement in the global frame is given by

$$y_{\text{global}}(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} X(k) + R(\theta)v(k), \tag{11}$$

where $R(\theta)$ is the rotation matrix between the local and the global coordinate systems given by

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}.$$

The term $v(k)$ in equation (11) represents the sensor noise. It has two components, range noise and the bearing noise. The range noise is usually smaller than the bearing noise. The range noise increases with the distance of the sensor from the target and the bearing noise variance can usually be modeled as a fixed multiple of the range noise variance for a given sensor. For simplicity, the two noises can be assumed independent. Thus the covariance matrix of $v(k)$ is typically given by

$$R = \begin{bmatrix} \sigma^2_{\text{bearing}} & 0 \\ 0 & \sigma^2_{\text{range}} \end{bmatrix},$$

where σ^2_{range} is the range noise variance that increases with the distance. The bearing noise variance $\sigma^2_{\text{bearing}}$ is related to the range noise variance for a particular sensor rather than the distance.

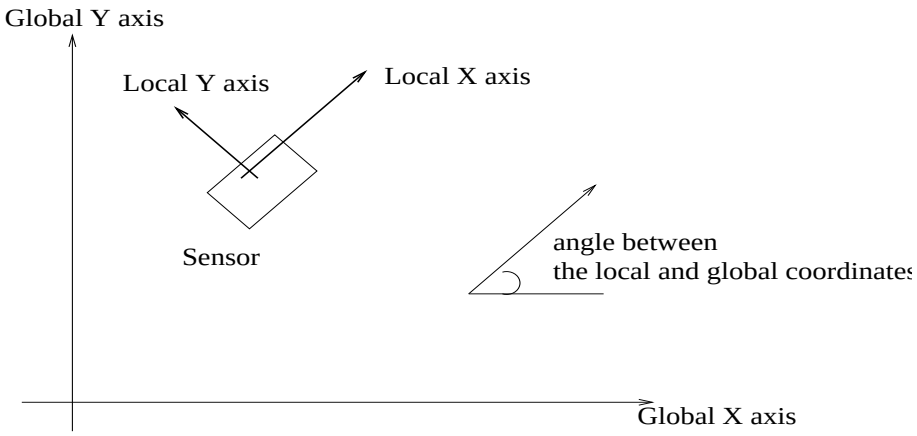

Fig. 2. If the sensor is oriented at an angle to X-axis, measurements need to be rotated to get their value in the global coordinates.

In the numerical example, we consider the value $h = 0.2$. The process noise is considered to have covariance matrix Q given by

$$Q = \begin{bmatrix} 1 & 0.25 \\ 0.25 & 1 \end{bmatrix}.$$

We consider two sensors. The first sensor is placed at position corresponding to $\theta = 0^\circ$ (see figure 3.) It is closer to the target and its bearing noise is assumed to be 6 times that of its range noise. The second sensor is at a position corresponding to $\theta = 90^\circ$ and its bearing noise is supposed to be twice that of the range noise. Specifically the numerical values of the sensor noise covariances considered are

$$R_1 = \begin{bmatrix} 2.4 & 0 \\ 0 & 0.4 \end{bmatrix} \qquad R_2 = \begin{bmatrix} 1.4 & 0 \\ 0 & 0.7 \end{bmatrix}.$$

Thus after rotation, R_1 remains the same while R_2 is transformed to

$$R_2 = \begin{bmatrix} 0.7 & 0 \\ 0 & 1.4 \end{bmatrix}.$$

We compare the algorithm performances over a time horizon of 20 steps. The cost function is simply the sum of the trace of the error covariance matrices of the two sensors from time $k = 0$ to time $k = 20$.

4.2 Choosing any one sensor always is not optimal

Note that the simple strategy of choosing the closer sensor (sensor 1) always is not optimal. We compare the strategy of choosing only sensor 1 or only sensor 2 with a randomly generated strategy that uses both the sensors with the sensor

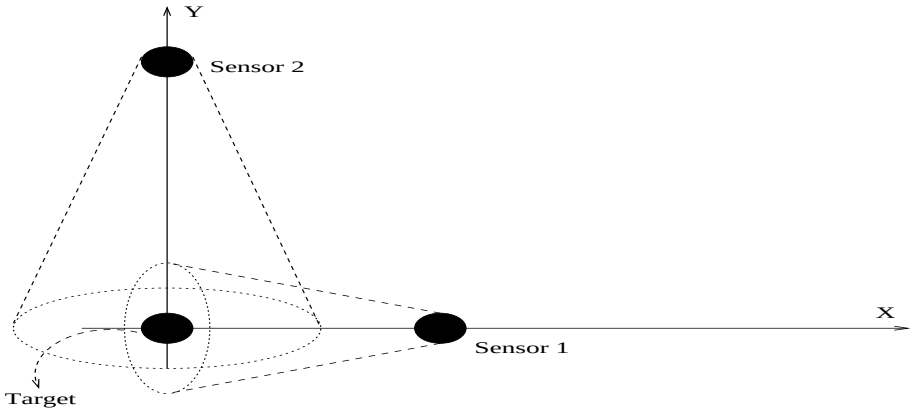

Fig. 3. Sensor orientation for the simulation examples.

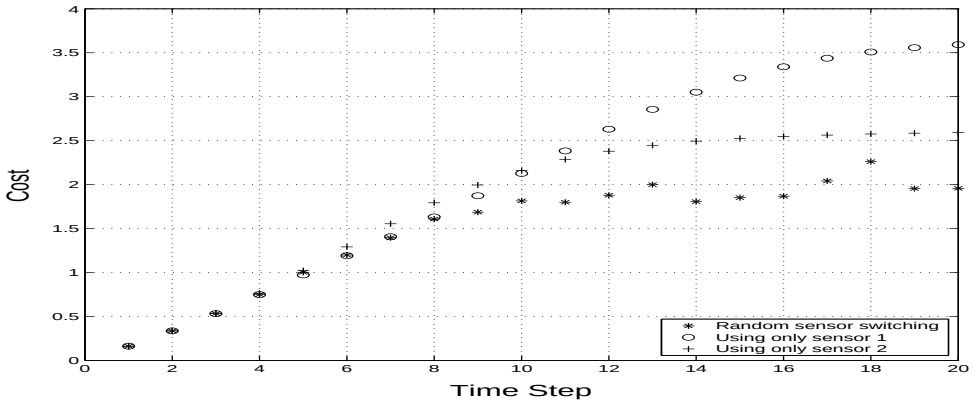

Fig. 4. Sensor switching helps to bring the cost down.

schedule [1,1,1,1,2,1,1,1,2,1,2,2,2,1,2,1,1,1,2,2] over the 20 time steps. The sum of traces of the error covariances of the two sensors for the three strategies as a function of time is shown in figure 4. We see that the even a random sensor switching strategy can help to bring down the cost. At any time step, the errors are much more if any single sensor is being used. In fact summed over the entire time horizon, we see that the switching strategy helps to bring down the cost by about 18% over any of the single sensor strategies.

4.3 Effect of communication noise

In this section, we consider the same example but add communication noise in the channel between the two sensors. The noise covariance is given by

$$R_{12} = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix}.$$

We consider the cost function as the sum of the traces of the error covariances of the two sensors over the time horizon $[0, 20]$. Figure 5 shows the improvement in cost by the sensor switching strategy given above over always using sensor 2 as the parameter α is varied. When α is small, the communication noise rapidly deteriorates the efficiency obtained by sensor switching since it deteriorates the estimates of both the sensors. As noted earlier, in the presence of communication

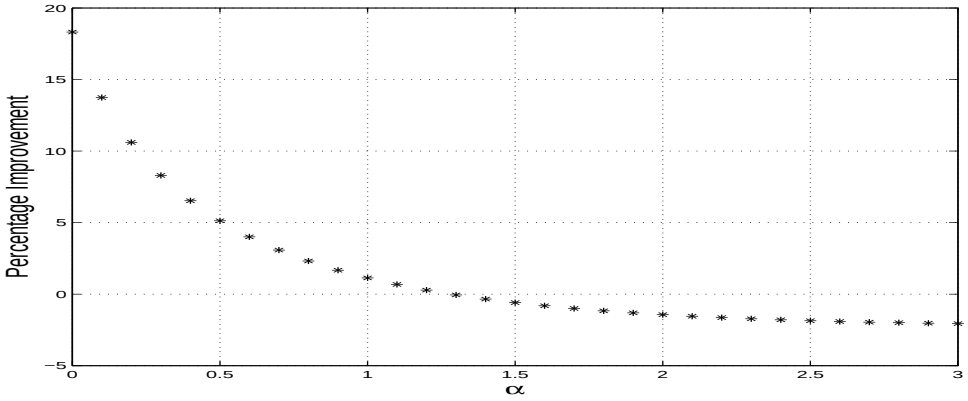

Fig. 5. Percent improvement in cost due to sensor switching as communication noise is increased.

noise, sending measurements might not be the optimal thing to do.

4.4 Performance of the sliding window algorithm

In this section we study the performance of the sliding window algorithm described earlier. We consider the same example and cost function as before. We find the optimal sequence for the noiseless case. Figure 6 shows the improvement in the cost due to the predicted (sub)-optimal sensor sequence over using only the sensor 2 as a function of varying window sizes. It can be seen from the figure that even a window size of $k = 1$ leads to more than 20% improvement in the cost by predicting a good sensor switching strategy.

4.5 Performance of the thresholding algorithm

We now consider the thresholding algorithm presented earlier for the same example and cost function as above. Figure 7 shows the improvement in cost due to the optimal sensor sequence predicted by the thresholding protocol as the cut-off factor f is varied. Again, a large improvement can be obtained by using a fairly small thresholding factor. For $f = 1$, the improvement is more than 20%.

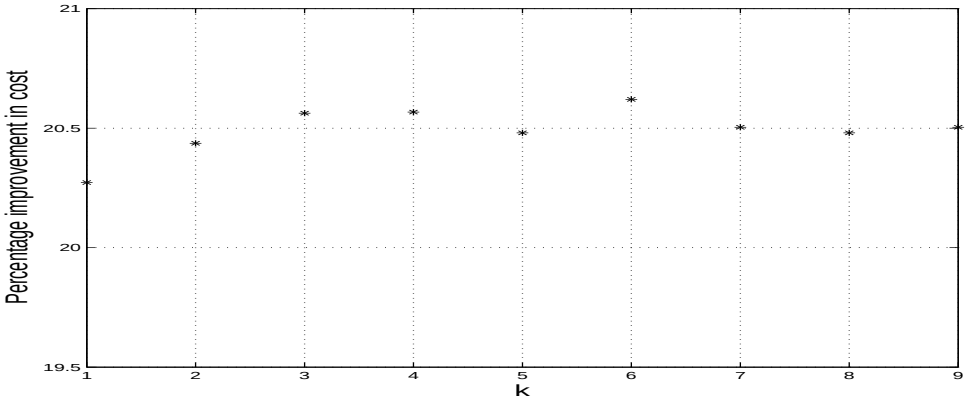

Fig. 6. Percent improvement in cost due to the optimal sensor switching strategy as predicted by the sliding window scheme.

4.6 Performance of the random choice algorithm

We find the optimal probability distribution for the random choice algorithm by optimizing the upper bound in equation (9) over q_1 and q_2. The optimal probability for sensor 1 turns out to be $q_1 = 0.41$. Indeed, if we find the optimal sequence by the thresholding algorithm, it turns out that in the steady state, the percentage of sensor 1 in the sequence is about 43%. For this probability distribution, the steady state value of the upper bound of the trace of the expected error covariance matrix for the two sensors turns out to be 2.1269, which compares well with the value of about 2 obtained by the optimal strategy given above.

5 Conclusions and Future Work

In this paper, motivated by the use of sonar range-finders on the vehicles on the Caltech MVWT, we looked at the problem of distributed estimation when only one sensor is allowed to take a measurement per time step. We saw that it is sufficient to exchange measurements between the sensors if the communication channel is noiseless and the optimal recursive estimation algorithm is simply a time-varying Kalman filter. We looked at how the performance degrades when there is communication noise present. Then we investigated the problem of determining an optimal sensor switching strategy. We saw that this problem involves searching a tree in general and proposed two strategies for pruning the tree to keep the computation limited. We also considered an algorithm which simply chooses sensors at random according to some probability distribution which can then be optimally chosen. Some examples demonstrating these algorithms were presented.

The work can potentially be extended in many ways. Obviously there exist better strategies for the case of communication noise, although they might entail

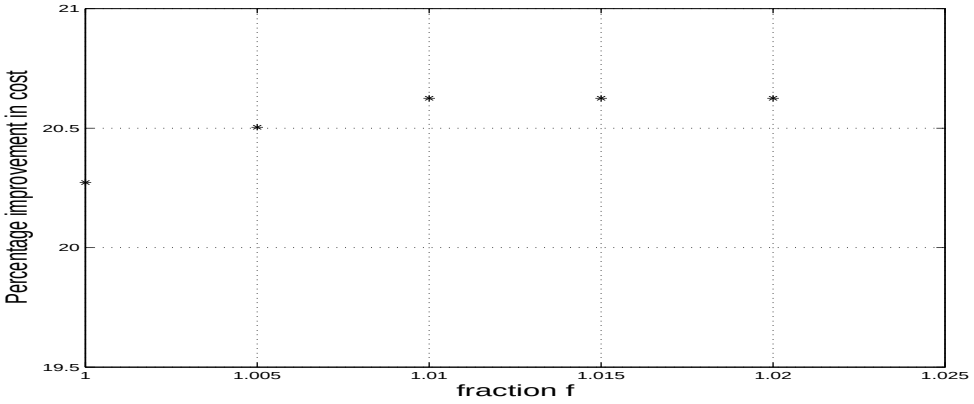

Fig. 7. Percent improvement in cost due to the optimal sensor switching strategy as predicted by the thresholding scheme.

more amount of data transmitted. Another avenue that can be investigated is the effect of data loss due to fading in the wireless channel.

Acknowledgements

The authors would like to thank Joel Burdick for helpful discussions. Research supported in part by the AFOSR grant F49620-01-1-0460 for the first author and by the Engineering Research Centers Program of the National Science Foundation under Award Number EEC-9402726 and also a grant from NASA for the second author.

References

1. S.I. Roumeliotis and G.A. Bekey, "Distributed multi-robot localization," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, Oct 2002.
2. J. Nash, "Optimal Allocation of Tracking Resource," in *Proceedings of the 1977 IEEE Conference on Decision and Control*, New Orleans, LA, December 1977, IEEE, vol. 1, pp. 1177–1180.
3. H. Karl, "Making sensor networks useful: Distributed services - the eyes project," ESF Workshop, La Spezia, Italy, 2002.
4. D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," in *Proceedings of the Fifth Annual Intl. Conf. on Mobile Computing and Networks*. MobiCOM, 1999.
5. J. Kahn, R. Katz, and K. Pister, "Next Century Challenges: Mobile Networking for "Smart Dust"," in *Proc. of ACM MobiCom Conference*, Seattle, WA, August 1999, MobiCom.
6. B.S.Y. Rao, H.F. Durrant-Whyte, and J.A. Sheen, "A fully decentralized multi-sensor system for tracking and surveillance," *Intl. Journal of Robotics Research*, vol. 12, pp. 20–44, 1993.

7. R. Mina and M. Bhardwaj and S.H. Cho and A. Sinha and E. Shih and A. Wang and A. Chandrakasan, "Low-Power Wireless Sensor Networks," *VLSI Design 2001*, January 2001.
8. A. Wang and A. Chandrakasan, "Energy Efficient System Partitioning for Distributed Wireless Sensor Networks," in *Proc. ICASSP 2001*. ICASSP, May 2001.
9. S. Dhillon and K. Chakrabarty and S. Iyengar, "Sensor Placement for Grid Coverage under Imprecise Detections," in *Proc. Intl. Conf. on Information Fusion*. FUSION, 2002, pp. 1571–1587.
10. S. Meguerdichian and F. Koushanfar and M. Potkonjak and M. Srivastava, "Coverage Problems in Wireless Ad-hoc Sensor Networks," in *Proceedings of IEEE InfoCom 2001*. IEEE Computer and Communication Societies, April 2001.
11. C. Shen, C. Srisathapornphat, and C. Jaikaeo, "Sensor information networking architecture and applications," *IEEE Personel Communication Magazine*, vol. 8, no. 4, pp. 52–59, August 2001.
12. L. Cremean, W. Dunbar, D. van Gogh, J. Hickey, E. Klavins, J. Meltzer, and R. M. Murray, "The Caltech multi-vehicle wireless testbed," in *Proceedings of the IEEE Conf on Decision and Control*, 2002.
13. A. Savkin, R. Evans and E. Skafidas, "The Problem of Optimal Robust Sensor Scheduling," in *Proceedings of the 39th Conference on Decision and Control*, Sydney, Australia, December 2000, CDC, number 1.
14. E. Skafidas and A. Nerode, "Optimal measurement scheduling in linear quadratic gaussian control problems," in *Proc. Of the 1998 IEEE Intl. Conf. On Control Applications*, Trieste, Italy, 1998, pp. 1225–1229.
15. G.A. McIntyre and K.J. Hintz, "An Information Theoretic Approach to Sensor Scheduling," in *Proceedings of the SPIE*, Orlando, FL, Apr 1996, Signal Processing, Sensor Fusion, and Target Recognition V, vol. 2755, pp. 304–312.
16. T. Kailath, A.H. Sayed, and B. Hassibi, *Linear Estimation*, chapter 9, Prentice-Hall, 2000.
17. Bo Lincoln and Bo Bernhardsson, "LQR optimization of linear system switching," *IEEE Transactions on Automatic Control*, vol. 47, pp. 1701–1705, Oct. 2002.
18. S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2003.
19. I. S. Gradshteyn and I. M. Ryzhik, *Tables of integrals, Series, and Products*, Academic Press, 2000.
20. B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry, "Kalman filtering with intermittent observations," in *Proceedings of the IEEE Conf on Decision and Control*, Dec 2003, To Appear.
21. Y. Bar-Shalom and X. R. Li, *Estimation and Tracking: Principles, Techniques and software*, Artech House, 1993.
22. K. Ramachandra, *Kalman filtering techniques for radar tracking*, Marcel Dekker Inc., New York, 2000.